M. L. Frankford EB33

CR-184103

Space Lab System Analysis

Advanced Solid Rocket Motor (ASRM) Communications Networks Analysis

Contract:
NAS8-36717

Final Report

12/31/90

Frank M. Ingels
Robert J. Moorhead II
Principal Investigators

Jane N. Moorhead
C. Mark Shearin
Dale R. Thompson
Research Assistants

Department of Electrical and Computer Engineering
Mississippi State University
Mississippi State, MS 39762
601-325-3912

submitted to:
NASA/MSFC
EB32/Suttle

Report Documentation Page

1. Report No.

2. Government Accession No.

3. Recipient's Catalog No.

4. Title and Subtitle

Space Lab System Analysis
Advanced Solid Rocket Motor (ASRM)
Communications Networks Analysis

5. Report Date

December 1990

6. Performing Organization Code

7. Author(s)

F. M. Ingels
R. J. Moorhead
J. N. Moorhead
C. M. Shearin
D. R. Thompson

8. Performing Organization Report No.

MSU-EE-FIN-30-90

9. Performing Organization Name and Address

Mississippi State University
Dept. of Electrical & Computer Engineering
Mississippi State, MS 39762

10. Work Unit No.

11. Contract or Grant No.

NAS8-36717

12. Sponsoring Agency Name and Address

NASA
Washington, DC 20546-0001

13. Type of Report and Period Covered

Final Report
Nov. 89 - Nov. 90

14. Sponsoring Agency Code

15. Supplementary Notes

Prepared by Mississippi State University for Gray Settle.

16. Abstract

A synopsis of research on computer viruses and computer security is presented. A review of seven technical meetings attended is compiled. A technical discussion on the communication plans for the ASRM facility near Iuka, Mississippi, is presented, with a brief tutorial on the potential LAN media and protocols.

17. Key Words (Suggested by Author(s))

ASRM - Advanced Solid Rocket Motor
LAN - Local Area Network
PSCN - Program Support Communications Network
Computer Virus, Computer Security

18. Distribution Statement

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of pages	22. Price
Unclassified	Unclassified	37	

Space Lab System Analysis

Advanced Solid Rocket Motor (ASRM) Communications Networks Analysis

Contract:
NAS8-36717

Final Report

12/31/90

Frank M. Ingels
Robert J. Moorhead II
Principal Investigators

Jane N. Moorhead
C. Mark Shearin
Dale R. Thompson
Research Assistants

Department of Electrical and Computer Engineering
Mississippi State University
Mississippi State, MS 39762
601-325-3912

submitted to:
NASA/MSFC
EB32/Suttle

TABLE OF CONTENTS

APPENDICES

1.0 INTRODUCTION

The Statement-of-Work (SOW) for Contract NAS8-36717, Option III was:

> The contractor shall continue providing technical expertise in networking distributed computer system architectures. Participation in design review meetings are required and independent critique shall be provided the Government. In addition to various system architectures being considered for the Marshall Avionics System Testbed (MAST), the contractor shall analyze the networks planned for the Advanced Solid Rocket Motor (ASRM) facility at Yellow Creek, Mississippi. This analysis shall consider estimated "worst case" data loading in the automated manufacturing of solid rocket motors. It is very important that the real-time aspect of manufacturing processes for (ASRM) be investigated and required safing functions be included in data loading.

The ASRM portion of the SOW was the main effort in the third and final year of this contract. Soon after Option III commenced, NASA decided to implement an interim facility near the Yellow Creek site, as opposed to launching immediately into the buildup of a permanent facility on the Yellow Creek site. Getting the interim facilities operational took precedence over designing the long-term, permanent plant, for which we were contracted to analyze the communication networks. Since it would therefore be a while before there was a network design to analyze, we were requested to look into the issues of computer security and computer virus. Our endeavors in this area are summarized in Section 2. For much more information, the reader is directed to the monthly reports covering the December 1989 to May 1990 time frame.

As the planning began to firm up on the network design, we began to redesign and rebuild our tools to analyze the proposed network designs in the appropriate ways. Our efforts in this area are summarized in Sections 3-5.

The five Appendices include:

- A list of the five interim final reports which have been submitted earlier.
- A list of the seven meetings we have attended over the past 13 months, which involved 29 person-days.
- A listing of the network analysis program which has been developed and some sample program runs.
- A list, albeit incomplete, of acronyms.

2.0 COMPUTER SECURITY AND VIRUSES

A lot of time was spent in the first half of 1990 learning about the security requirements for ASRM and about viruses. A brief summary of computer viruses is in section 2.1. In section 2.2 the Internet Worm, propagated by Robert Morris, and some potential ways of protecting against such worms are described. In section 2.3 we include a bibliography of papers on computer viruses and the Internet worm. A copy of each reference can be provided on request.

2.1 COMPUTER VIRUSES

There are many types of malicious software: Trojan horses, seemingly performing benign actions while also performing covert actions; Logic Bombs, performing some pre-programmed action when given the appropriate trigger ; Worms, crawling through computer networks, frequently changing their location to avoid detection, and taking over the resources of a computer, using it for their own purposes; and Viruses, attacking other system files, modifying them so that they contain a copy of the virus. All of these forms of malevolent software pose a considerable risk to computers, whether linked to a network or not. Viruses can be the most dangerous form because they often exhibit many of the attributes of Trojan Horses, Logic Bombs and Worms. Even viruses written as harmless pranks can be harmful, using more memory than their writers intended and causing unintended boot failures and system crashes. Most viruses work on an operating system dependent manner. Thus there are IBM PC DOS viruses, Macintosh viruses, and Unix viruses to name a few. There is currently a big push toward developing ways to diagnose and stop viruses before they infiltrate PCs and networks.

Several types of antiviral programs are available. Obviously, these packages can only work for known viruses so none are totally virus perfect. Integrity checkers generate a checksum for each program on your system. At boot-up time, these values are compared against the stored values for each program. This works for programs that don't use their .COM files to store configuration options set by the individual.

Monitoring programs sit on the machine interrupts and examine each for suspicious activity. Virus removers examine the hard disk for signs of viruses and alert the user if suspicious activity is recognized. Some of the programs even remove the virus. These programs can only work for those viruses that are in existence today and that the author of the software or hardware has incorporated into the code.

For stand-alone PCs, the main input avenue for a virus is through infected floppy disks. Even new software is susceptible to carry a virus. CD_ROMs have also been known to carry viruses.

Many people use backups to protect against viruses. This certainly is not a fool-proof method but does provide some protection. Suggestions for backing up material includes backing up only data files and rotating through several different backup copies, e.g. one for each day of the week. Unfortunately viruses often have triggers that allow the virus to remain dormant for long periods before becoming active. Thus backups could contain the virus unknowingly.

Other preventive measures against viruses include always putting write-protect tabs on non-data disks, and not loaning out program disks. On the hard disk, make all .COM and .EXE files read-only. This can be easily done with the ATTRIB command. Move the COMMAND.COM file out of the root directory; many viruses work specifically on this file. Don't let others use your system. This can be done by locking up the system with keys.

Perhaps the most sensible advice is to use as many preventive measures as possible but don't be fooled into thinking you can beat the threat of a possible virus.

2.2 THE INTERNET WORM

Although the worm that entered the Internet system on November 2, 1988, did not destroy any files, intercept private mail, reveal passwords, corrupt databases, or plant Trojan horses, it certainly opened the public's eye to how weak the computer security defenses really are. The worm, only infecting Sun 3 systems and VAX computers running variations of 4.X BSD

Unix, took advantage of some flaws in standard software installed in many Unix environments.

One flaw in the Unix system that allowed the migration of the worm is the meager password security. Users passwords are encrypted using a permuted version of the Data Encryption Standard (DES) algorithm. This algorithm is assumed to be unbreakable because the exhaustive search would require massive amounts of time. Yet new and faster technology has changed the attitude concerning the invincibility of the DES algorithm. Also many passwords are trivial and easy to guess. By using parallel checking and trying easily–guessed passwords, the worm was able to access over 50% of the accounts at some sites.

The current password utility makes only a minimal attempt to ensure new passwords are non-trivial to guess. Many of the passwords used by the worm were merely the user id or some perturbation of it. The worm also used the online dictionary as a source for possible passwords. Policies should be put in place to require a minimum number of letters, a minimum combination of numbers and letters, uppercase as well as lower case, not based on account name or in the online dictionary.

The password of each user, although encrypted, is stored in a publicly readable file. The file also stores accounting information; thus a user can utilize the file to obtain information about other users. By organizing the password with publicly available data, an attacker could have a list of possible passwords, encrypt them, and compare them with the actual passwords without any attention flag being raised. If the encrypted passwords were stored in a shadow password file only readable to system administrators, outside intrusion could be prevented. Also, the encryption technique and the command to compare encrypted words should be privileged commands. Additionally, a threshold could be set to check for repeated attempts from the same process.

The VAX/VMS System Security Guide that comes with VMS should be read by system managers. There are techniques included that might prevent break ins. Also there is a security toolkit put out by SPAN–NSN (Space Physics Analysis Network– NASA Science Network) that should be looked into.

2.3 BIBLIOGRAPHY

Brar, Jay, "Microcomputer Viruses: A Threat Computing Facilities Cannot Afford to Ignore," JPL Computing and Information Services News, January 1990, Vol.8, No.1, pp. 1–6.

Greenberg, Ross M., "Know Thy Viral Enemy," Byte, June 1989, pp. 275–280.

Highland, Harold, "Highland on Hackers: Common Sense Before Legislation," IEEE Spectrum, February 1990, pp. 50.

McAfee, John D., "Managing The Virus Threat," Computerworld, February 13, 1989, pp. 89–95.

Rawles, James W., "The Viral Threat," Defense Electronics, February 1990, pp. 62–67.

Rounds, Frederic N., "NSI's Management Perspective of its Evolving Networks," NASA OSSA Information Systems Newsletter, November 1989, Issue 18, pp. 34–35.

Rubenking, Neil J., "Ten Simple Steps to Virus Protection," PC Magazine, April 25, 1989, pp. 195.

Seeley, Donn, "Password Cracking: A Game of Wits," Communications of the ACM, June 1989, Vol. 32, No. 6, pp. 700–703.

Sisson, Pat, "The Space Physics Analysis Network Security Toolkit: Tracking System Vulnerabilities," NASA OSSA Information Systems Newsletter, July 1989, pp. 45–46.

Spafford, Eugene, "The Internet Worm: Crisis and Aftermath," Communications of the ACM, June 1989, Vol. 32, No. 6, pp. 678–687.

Stephenson, Peter, "Personal And Private," Byte, June 1989, pp. 285–288.

Weiner, Daniel P., "When a Virus Makes Your PC Sneeze," US News & World Report, February 26, 1990, pp. 62.

3.0 LOCAL AREA NETWORKS USING OPTICAL FIBERS

Even though optical fibers are more expensive than the standard copper medium of transmission (twisted pair, coaxial, baseband), using fiber optics in a LAN offers several distinct advantages:

- Because they use light instead of electricity, fiber optic cables are free from electromagnetic interference, crosstalk, and other types of noise except that which is introduced into the system from the electronic interfaces to the network. This is especially useful for sites with high levels of EMI.
- Since they have a large bandwidth with little inherent loss, optical fibers can provide data rates up to around 100 Gbits/s over 100km [KLT87]. One reason for this is that the bandwidth is inversely proportional to the length, while with wire the bandwidth is inversely proportional to the length squared.
- Due to the fact that taps are difficult to place in the network, optical fibers are very secure from unwanted intrusion.
- Since they are physically small and lightweight, optical fibers aid installation and maintenance. An optical fiber is generally 1/6 the weight of an equivalent coaxial cable carrying the same amount of information.
- Because optical fibers currently propagate with very little attenuation - typically as low as 0.2 dB/km - repeaters are not necessary for distances under 100km [JO88].
- Since optical fibers carry no electrical current, they are ideal in situations where a spark could set off volatile substances.

3.1 TOPOLOGIES

The selection of the topology, or physical layout/wiring of the LAN is one of the most important design considerations. Each of the four basic types of LAN topologies have pros and cons that must be weighed to select the proper topology for the specific needs of the network.

3.1.1 Bus

In the bus topology, the network is connected linearly, as can be seen in Figure 3.1.1. Each node along the line has access to the channel. The message is sent with an addressing code so that only the proper node will interpret the message.

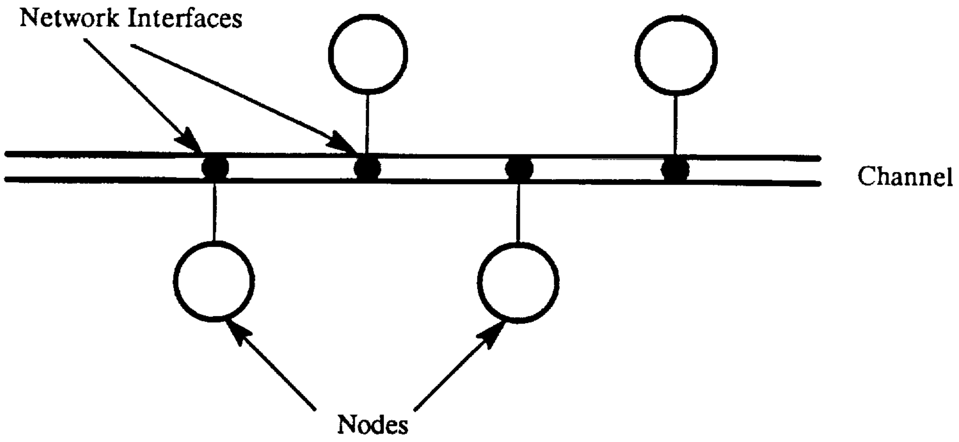

Figure 3.1.1: Bus Topology Diagram

To implement the bus topology on a fiber optic medium, either a passive or active coupling interface may be used (see Figures 3.1.2 and 3.1.3). At every active network interface, the entire optical signal enters the interface, undergoes an opto–electronic (denoted as O/E in the figure) conversion, is run through the node, and then is re–inserted back into the fiber medium through an electo–optic (E/O in the drawing) converter. The disadvantage of the active network is the cost and complexity. The passive network interface sends and receives messages directly from the fiber medium without the need for a break or interruption in the fiber. The passive network is a lossy implementation, and is limited to about 13 network interfaces that may be cascaded in sequence [KLT87].

A special case of the bus configuration is the tree. This is a more complex bus form that branches off from the common communication channel. The network branches are usually optical hubs.

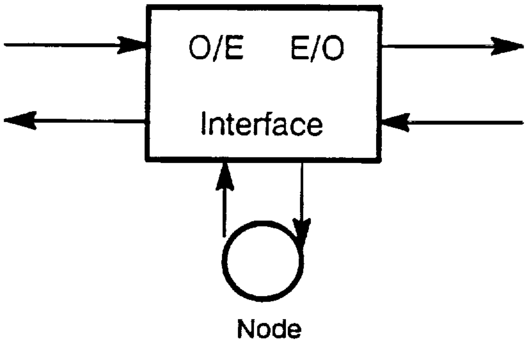

Figure 3.1.2: Linear Bi–directional Active Bus Network Interface

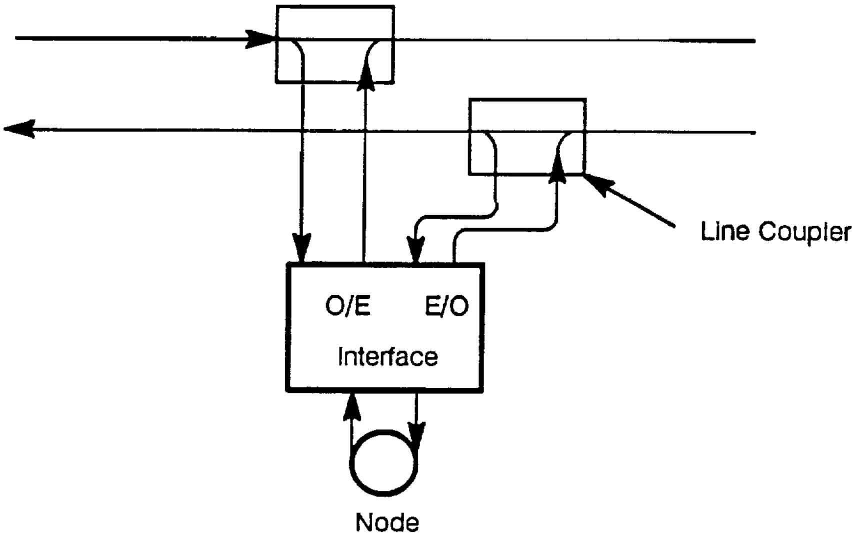

Figure 3.1.3: Linear Bi–directional Passive Bus Network Interface

3.1.2 Ring

The ring topology (see Figure 3.1.4) is distinct from the bus configuration in that the nodal arrangement forms a closed, roughly circular layout. Like the bus topology, all the nodes on the ring receive the message and decide whether to interpret it or not based on the destination address. Also like the bus arrangement, there are active and passive ring systems.

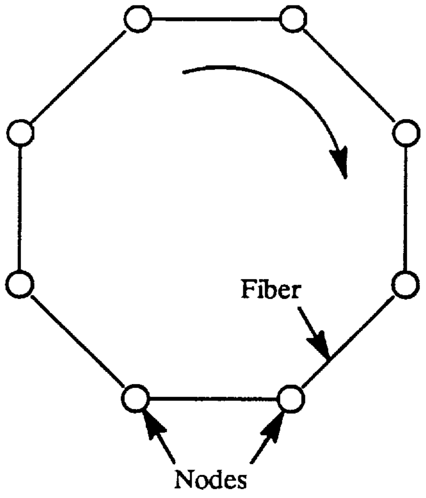

Figure 3.1.4: Ring Topology Diagram

In the active ring configuration, each node converts all of the optical energy to electrical signals to analyze it. This procedure serves a secondary function as a repeater. While the active ring has a relatively simple design with high-performance, they are generally costly and difficult to service, since every node failure or fiber break will crash the whole network.

In the passive ring configuration, each node derives its message from the optical fiber without interrupting the signal for opto-electronic conversion. This means that a node failure would not always crash the network like the active case, but a fiber break still will. This method brings its own set of problems, however. Because there is a continuous loop with no regeneration (opto-electric to electro-optic conversions at each node), echoes and unwanted distortions will arise in the medium. A signal that is injected into the fiber will remain there until attenuation eventually removes it.

3.1.3 Star

The star topology (see Figure 3.1.5) consists of a centralized hub that is linked to every other hub or node in the system point-to-point. The centralized hub also switches between the various nodes giving collision detection and CSMA capabilities. At its best, a star topology makes use of simple, point-to-point fiber connections without any problems of tap losses or echoes that the other two types of topologies have. At its worst, however, it has other problems like complex medium-access protocols and extensive collision detection and switching equipment.

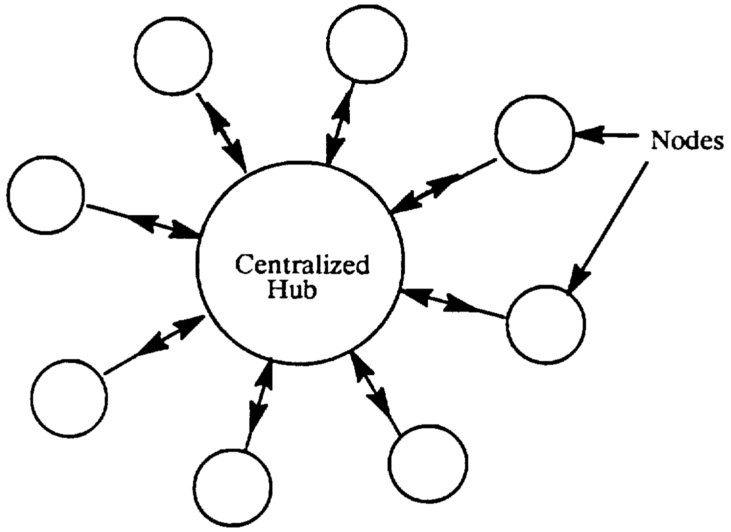

Figure 3.1.5: Star Topology

Like the ring and bus topologies, the star can be either passive or active. Because of the similarities to an Ethernet broadcast bus, the passive star can have CSMA/CD protocols with little modifications. However, since the passive star does not include regeneration, there may be different optical signal power levels that are received by nodes in the system that arise from collisions. To correct this problem, a separate collision sensing device may be externally coupled to the passive star coupler.

3.2 PROTOCOL TYPES

3.2.1 Ethernet

The Ethernet system was designed by XEROX and uses carrier-sense multiple-access with collision detection (CSMA/CD) [IN88]. Many different stations are connected to a common bus. If the bus is silent then the station will try to transmit a packet of data and then wait for an acknowledgement (ACK) from the receiving station. Once the receiving station sends an ACK then the transmitting station will send another packet. If two stations try to transmit at the same time then the information will collide. Then each station waits a random amount of time before trying to transmit again. If the station collides again then it waits a longer time. Each station has an exponential backup algorithm so the more collisions the longer each station will wait and the bus will quiet down. As long as the information is bursty in nature then the system will work.

The data rate for Ethernet is 10 Megabits per second (Mbps). The maximum station separation is 2.5 km. A maximum of 1024 stations can be connected to one Ethernet system. Normally, coax cable is used to interconnect the computers although fiber optical fibers can be used now. The standard topology is a bus topology.

On the data link layer, Ethernet uses CSMA/CD. The data is sent in variable size frames commonly called "packets" with a minimum spacing of 9.6 μs. The frame construction consists of:

1. 64 bit preamble
2. 48 bit destination address
3. 48 bit source address
4. 16 bit type field
5. 46 to 1500 bytes data field
6. 32 bit CRC error check field

The preamble provides synchronization and frame mark. The destination address contains the physical addresses of a particular station or a group of stations. The source address contains the physical address of the transmitting station. The type field is used by high–level network protocols. The data field contains the data being sent. The error check field consists of a Cyclic Redundancy Check which is generated by the transmitting station. The receiving station generates a CRC when it receives a packet and checks it against the received CRC. If they do not match then the transmission was garbled and the receiving station will ask for the packet again. This continues until an ACK is received and then the transmitting station can send another packet. Ethernet allows 15 re–tries before the station times out [IN88].

3.2.2 FDDI

FDDI, or Fiber Distributed Data Interface, is a network standard developed by the American National Standard Institute (ANSI X3T9.5) that runs at 100Mbps, or around ten times the speed of Ethernet. FDDI was started as a high–speed network for provide packet data between processors and fast storage devices. Now, FDDI can also be used as a high–speed low–error rate backbone to interconnect slower LAN's like IEEE 802.3 (CSMA/CD), 802.4 (Token Bus), or 802.5 (Token Ring). FDDI uses optical fibers for the communication medium and a timed token media access protocol with bus topology which provides each node equal access to the network. For the transmitting devices, Light Emitting Diodes (LEDs) are generally used. By using multi–mode optical fibers, links around 2 km are standard [KLT87]. By using single–mode optical fibers with laser diode transmitters the link distance can be extended up to 60 km. FDDI has several distinct advantages:

- Up to 1000 connections.
- Total fiber path length up to 200 km.
- Bit error rate (BER) less than 2.5E–10 [JA90]

FDDI uses a form of serial baseband transmission that combines both the data and the clock transmissions on a single bit stream. Because the clock information is transferred with the data, synchronization is accomplished with the recovery of the data.

FDDI can use Manchester encoding, like Ethernet, but normally FDDI uses 4b/5b with NRZI encoding. 4b/5b means that it uses combinations of five code bits to represent a symbol of four bits. NRZI is an edge–type code that is short for "Non–Return to Zero Invert on ones" – which in optical fibers deals with polarity transitions. Every polarity change results in a logical "0" (low) while no change in polarity results in a logical "1" (high). Manchester encoding, on the other hand, is a level–type code where a "zero" starts at logic low and makes a low to high transition in the middle of a clock cycle, and a "one" starts at logic high and makes a high to low transition in the middle of a clock cycle. The 4b/5b NRZI coding is chosen over the standard Manchester encoding system that Ethernet uses for two major reasons:

- The 4b/5b with NRZI encoding is more efficient, requiring cheaper components.
- Along with the frame formats, the 4b/5b with NRZI encoding allows easier detection and correction of errors.

The 4b/5b encoding scheme is more efficient in that it converts four data bits to five code bits, resulting in an 80% efficiency. This would require the optical components to be 125 Mbps in order to obtain the standard 100 Mbps required for FDDI. With the Manchester encoding scheme there are two pulses per data bit resulting in a 50% efficiency that would require 200 Mbps components for the system to run at the standard 100Mbps.

The manufacturing data network proposed for the Yellow Creek site is a hybrid of FDDI and IEEE 802.3/Ethernet. The connections from building to building will be Ethernet/IEEE 802.3 based, while within the OIS building, FDDI will link the five Ethernet lines together for processing in the central VAX computer. All of the fibers installed in the system will be FDDI compatible 62.5 micron fibers to easily changeover to a full FDDI system, should the need arise.

3.3 PERFORMANCE MEASUREMENTS

The performance of a LAN network can be broken into four different parts:

1. Response time
2. Delivery time
3. Throughput rate
4. Accuracy

The response time is the amount of time that the system will respond to a command issued by the transmitting station. The delivery time is the amount of time that a message will take to arrive at the receiving station. The throughput rate is the percentage of packets that are received without errors. The accuracy is a measure of how many errors the system will introduce.

4.0 COMPUTER SIMULATION OF A LAN

The throughput of a LAN is the percentage of packets that are received without errors. To simulate the throughput of the LAN at Yellow Creek a program was written in C using a modified version of the flow chart found in Telecommunications and Data Communications System Design with Troubleshooting by Harold B. Killen [KIL86]. The program uses a Poisson distribution and assumes a constant service time. It does not take into account the exponential backoff of the nodes during collision detection.

The configuration for the LAN is loaded as a separate data file. The parameters of the particular LAN are in a separate data file and loaded by the user. The parameters are:

q = the quitting point
nn = number of nodes
d = average distance between nodes in feet
bp = bits per page
ps = pages per second
bc = bits per packet
pf = propagation factor of the media
ts = transmission speed in bits per second
ls = speed of light in feet per second

To simplify the program all times are put in terms of slot times. At the end of the program all slot times are converted back to seconds. The program calculates the data entry rate (de) in packets per second, the slot time, and the propagation delay. Then it calculates the interarrival time assuming a Poisson distribution.

at = –aa*ln(P)
aa = mean arrival time
P = random number between 0 and 1

If the interarrival time is less than the propagation delay then a collision occurs. If the amount of time for a packet to arrive is less than the interarrival time then a collision occurs. The number of successful packets and collisions is counted along with the time. Then the throughput is calculated and compared to the theoretical throughput predicted by the following formula [KIL86]:

$$Th = G\exp(-AG)/(G(1+2A)+\exp(-AG))$$

where G = total packets per slot time

sl = slot time = number of bits being sent / the transmission speed

A = propagation delay in slots

The first LAN configuration consists of one node talking to the main computer on the CASE PREP LAN trying to send one page of text. A distance of one thousand feet was chosen because it is the largest distance between nodes in CASE PREP. The number of bits per page was chosen assuming 25 lines of 80 characters which gives 2000 characters. Assuming standard ASCII of 8 bits this gives 16000 bits per page. The number of pages per second was chosen to be 1 page per second as given in the original specifications. The largest packet consists of 1518 bytes or 12144 bits. An average index of refraction of 1.46 was used to calculate the propagation factor using $v = c/n$. Therefore the propagation factor is $1/n$. The transmission speed of 10 Megabits per second was chosen since this is the speed that the Ethernet protocol uses.

The results of this configuration are shown in Appendix D. The throughput is low because throughput is defined as the average usage time without conflicts divided by the summation of average idle and busy times.

The throughput is $S = U/(B + I)$, where

U = average usage time without conflicts

B = average busy time

I = average idle time

Since there was a large idle time, the throughput is low (.0016). The LAN efficiency factor is the throughput divided by the offered load. It tells what percentage of time the user will find the system open for use or not busy. In this case, the channel will be open for use 99.16% of the time.

The second LAN configuration simulates the same LAN, but with 24 nodes trying to send one page of text. The throughput is still low because the LAN is lightly loaded. The LAN efficiency decreased to 96.72%.

The third LAN configuration consists of one node talking to the main computer on the CASE PREP LAN trying to send one page of graphics. A distance of one thousand feet was chosen because it is the largest distance between nodes in the CASE PREP. The number of bits per page was chosen assuming 640 by 480 pixels with 4 bits per pixel which gives 1,228,800 bits per page. The number of pages per second was chosen to be .3 pages per second which gives each node 3 seconds to transmit one graphics page as given in the original specifications. The throughput increased to .0356. The LAN efficiency dropped to 93.12%.

The fourth LAN configuration consists of 24 nodes talking to the main computer on the CASE PREP LAN trying to send one page of graphics. A distance of one thousand feet was chosen because it is the largest distance between nodes in the CASE PREP. The number of bits per page was chosen assuming 640 by 480 pixels with 4 bits per pixel which once again gives 1,228,800 bits per page. The number of pages per second was chosen to be .3 pages per second which gives each node 3 seconds to transmit one graphics page as given in the original specifications. The theoretical throughput increased to .2276 while the simulated throughput increased to .1857. The LAN efficiency dropped to 62.98%. Therefore 62.98% of the time the user will find the network open to use. This also means that 37.02% of the time the user will find the network busy.

5.0 REFERENCES

[KIL86]

Killen, Harold B., Telecommunications and data communications system design with troubleshooting, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1986.

[IN88]

Ingles, F. M., "Local Area Networks", 1988

[JO88]

Jones, W. B. Jr., Introduction to Optical Fiber Communication Systems, Holt, Rinehart and Winston, Inc., New York, New York, 1988.

[KLT87]

Kummerle, Karl, Fouad A. Tobagi, and John O. Limb, Advances in Local Area Networks, The Institute of Electrical and Electronics Engineers, Inc., New York, New York, 1987.

[JA90]

Jain, Raj, "Error Characteristics of Fiber Distributed Data Interface (FDDI)", IEEE Transactions on Communications, August 1990, Vol. 38, No. 8, page 1244–1252.

APPENDIX A

Interim Final Reports
Contract Number NAS8-36717

Title and Subtitle	Period Covered	Performing Organization Report Number	No. of Pages
Spacelab System Analysis	9/86 to 9/87	MSU-EE-FIN-9-87	182
Spacelab System Analysis	9/87 to 9/88	MSU-EE-FIN-9-88	214
Spacelab System Analysis-Marshall Avionics Systems Testbed (MAST)	6/87 to 10/88	MSU-EE-FIN-10A-88	213
Spacelab Systems Analysis-A Study of Communications Systems for Advanced Launch Systems	6/87 to 10/88	MSU-EE-FIN-10B-88	209
Spacelab System Analysis-The Modified Free Access Protocol: An Access Protocol for Communication Systems with Periodic and Poisson Traffic	10/88 to 10/89	MSU-EE-FIN-20A-89	248
Spacelab System Analysis-Marshall Avionics Systems Testbed (MAST)	10/88 to 10/89	MSU-EE-FIN-20B-89	35

APPENDIX B

Meetings attended 11/01/89 through 11/30/90

1. Nov. 6–7, 1989	Communications Working Group 3 people, at NASA/MSFC
2. Jan. 7–8, 1990	Communications Working Group 2 people, at NASA/MSFC
3. March 27–29, 1990	Full Review of AIMS Specification 3 people, at NASA/MSFC
4. July 19, 1990	status/update meeting 2 people, at RUST International
5. July 25–27, 1990	OIS Computer Specification Review 2 people, at NASA/MSFC
6. September 14, 1990	Communications Working Group 3 people, at NASA/MSFC
7. November 16, 1990	Pre–review of OIS Communications Specs 1 person, at NASA/MSFC

APPENDIX C
PROGRAM LISTING

```
/**********************************************************/
/* This program simulates a Local Area Network.           */
/* It uses a Poisson distribution and assumes a constant   */
/* service time.                                          */
/**********************************************************/

/**********************************************************/
/* Definition of variables.                               */
/**********************************************************/
/* q = quitting point                                     */
/* nn = number of nodes                                   */
/* d = distance between nodes in feet                     */
/* bp = bits per page                                     */
/* ps = pages per second                                  */
/* bc = bits per packet                                   */
/* pf = propagation factor                                */
/* ts = transmission speed in bits per second             */
/* ls = speed of light in feet per second                 */
/*                                                        */
/* de = data entry rate in packets per second             */
/* sl = slot time in seconds (unit used in program)       */
/* pd = propagation delay in slots                        */
/* aa = average arrival time in slots                     */
/* at = arrival time in slots                             */
/* u = channel usage without conflicts in slots           */
/* tu = total channel usage w/o conflicts in slots        */
/* s = sl + pd total number of slots for message          */
/* b = busy channel time in slots                         */
/* tb = total busy channel time in slots                  */
/* i = idle time in slots                                 */
/* ti = total idle time in slots                          */
/* to = total channel usage in slots                      */
/*                                                        */
/* ol[] = offered load in slots                           */
/* ab = average bus busy time in slots                    */
/* au = average channel usage without conflicts in slots  */
/* ai = average idle time in slots                        */
/*                                                        */
/* ss[] = simulated throughput                            */
/* st[] = theoritical throughput                          */
/* ot = operating time in slots                           */
/*                                                        */
/* top = temporary variable                               */
/* bot = temporary variable                               */
/* x = counter for number of nodes talking                */
/* z = counter                                            */
/* c = cycle counter                                      */
/* co = number of collisions                              */
/* a = number of successful packets                       */
/* tc = total number of collisions                        */
/**********************************************************/

/* Load in data file of the LAN network                   */

#include <math.h>
#include <fcntl.h>
#include <stdio.h>

void csma ();
void statistics ();
void statscsma ();
void theory ();
void output ();
void printout ();
void initial ();
```

```
FILE *datafile, *outputfile;

/* Declare variables */
int q,nn,x,z,c,co,nb,a,tc[1000],c1[1000];

float d,bp,ps,de,bc,pf,ts,ls,sl,pd,aa,at,u,tu,s,b,tb,i,ti,to,ol[1000],ab,au,ai,ss[1

/*********************************************************/
/* The main computes the slot time and propagation delay  */
/* It also calculates the arrival time using Poisson      */
/* distribution.  A random number between 0 and 1 is      */
/* generated using rand().  It then checks for a collision*/
/*********************************************************/

main ()
{
        char in[80];
        char out[80];
        double temp1, temp2;
/* Input lan configuration */
        printf("\n");
        printf("What file next? ");
        scanf("%s",in);
        getchar();
        datafile = fopen(in,"r");
        strcpy(out,in);
        strcat(out,".out");
        outputfile = fopen(out,"w");

/* Load in data file of the LAN network  */
        fscanf(datafile,"%d",&q);
        fscanf(datafile,"%d",&nn);
        fscanf(datafile,"%e",&d);
        fscanf(datafile,"%e",&bp);
        fscanf(datafile,"%e",&ps);
        fscanf(datafile,"%e",&bc);
        fscanf(datafile,"%e",&pf);
        fscanf(datafile,"%e",&ts);
        fscanf(datafile,"%e",&ls);
        close(in);

/* Start calculating */
        de = bp*ps/bc;
        sl = bc/ts;
        pd = d/sl/(pf*ls);
        for (x = 1; x <= nn; x = x +1 )
        {
        at=co=c=ti=tu=a=b=tb=to=0;
        for (z = 1; z <= q; ++z)
        {
            c = c + 1;
            aa = 1.0/(de*(float)x)/sl;
            temp1 = rand();
            temp2 = temp1/2147483647;
            at = -aa*log(temp2);
            csma ();
         }
         statistics ();
        }
        initial ();
        output ();
        printout ();
}
/*********************************************************/
/* Collision sense multiple access.  Checks for collision */
/*********************************************************/
```

```
void csma ()
{
        u = 1;
        s = 1 + pd;
/***********************************************************/
/* Check if arrival time less than propagation delay       */
/***********************************************************/
        if (at<=pd)
        {
          co = co +1;
          u = at;
          b = u + 1 + pd;
        }
/***********************************************************/
/* Check if amount of time for packet to arrive is less    */
/* than the arrival time predicted by Poisson.             */
/***********************************************************/
        if (s<=at)
        {
          b = s + u;
          tb = tb + b;
          i = at - s;
          ti = ti + i;
          a = a + 1;
          tu = tu + u;
          to = to + b + i;
        }
}
void statistics ()
{
        nb = nb + 1;
        cl[nb] = co;
        statscsma ();
        tc[nb] = c - a;
        ol[nb] = de*(float)x*sl;
        theory ();
}
void statscsma ()
{
        if (a!=0)
        {
          ab = tb/a;
          au = tu/a;
          ai = ti/a;
          ss[nb] = au/(ab+ai);
        }
}
/********************************************************/
/* Calculates the theoritical throughput                */
/********************************************************/
void theory ()
{
        float top;
        float bot;
        top = ol[nb]*exp(-(pd*ol[nb]));
        bot = ol[nb]*(1+2*pd) + exp(-(pd*ol[nb]));
        st[nb] = top/bot;
}
/*******************************************************/
/* Output the results                                  */
/*******************************************************/
void output ()
{
        /*convert slots back to seconds*/
        x = x-1;
        ot = (float)q/((float)x*de);
        to = to*sl;
        ai = ai*sl;
        ab = ab*sl;
        au = au*sl;
        ti = ti*sl;
        tb = tb*sl;
        tu = tu*sl;

        printf("\n");

        /*print out stats for number of nodes trying to talk*/
        printf("TIME COMPONENTS OF SIMULATION \n");
        printf("Number of Nodes =  %d \n",x);
        printf("Total operating time (seconds)             %12.8f \n",ot);
        printf("Total channel usage (seconds)              %12.8f \n",to);
        printf("Average idle time (seconds)                 %12.8f \n",ai);
        printf("Average busy time (seconds)                 %12.8f \n",ab);
        printf("Average usage time w/o conflicts (sec)      %12.8f \n",au);
        printf("Total idle time (seconds)                  %12.8f \n",ti);
        printf("Total busy channel time (seconds)           %12.8f \n",tb);
        printf("Total channel usage time w/o conflicts      %12.8f \n",tu);

        printf(" \n");
        printf(" \n");
}
void initial ()
{

        /***************************************************/
        /* printout initial conditions                     */
        /***************************************************/
        printf("\n");
        printf("            LAN SIMULATION \n");
        printf("**************************************** \n");
        printf("\n");
        printf("INITIAL CONDITIONS \n");
        printf("Quitting point                             %12d \n",q);
        printf("Number of nodes                            %12d \n",nn);
        printf("Distance between nodes in feet =           %12.0f \n",d);
        printf("Bits per page =                            %12.0f \n",bp);
        printf("Pages per second =                         %12.2f \n",ps);
        printf("Bits per packet =                          %12.0f \n",bc);
        printf("Propagation factor =                       %12.2f \n",pf);
        printf("Transmission speed in bits per second      %e \n",ts);
        printf("Speed of light in feet per second          %e \n",ls);
        printf("\n");
        printf("CALCULATED CONDITIONS \n");
        printf("Data in packets per second =               %12.8f \n",de);
        printf("Slot time in seconds =                     %12.8f \n",sl);
        printf("Propagation delay in seconds =             %12.8f \n",pd);
        printf(" \n");
        printf(" \n");
}
void printout ()
{
        float ef = 0;
        int j =0;
        float sps;
        /*convert slots back to seconds*/
        pd = pd*sl;

        /* Printout results and throughput */

        printf("RESULTS OF SIMULATION \n");
/*      for (j = 1; j <= nb; j++) */
/*      {                          */
/* Only printout largest number of nodes talking */
        j = nb;
        ef = ss[j]/ol[j];
        printf("Number of nodes talking %d \n",j);
        printf("Theoritical throughput (ST)                       %3.4f \n",st[j]);
        printf("Simulated throughput (SS)                         %3.4f \n",ss[j]);
        printf("Total number of collisions                        %d \n",tc[j]);
        printf("Offered load in packets/slot    (G)               %3.4f \n",ol[j]);
        printf("LAN efficiency factor (SS/G)                      %3.4f \n",ef);
        printf(" \n");
        printf("\n");
/*      }    */

}
```

APPENDIX D
SAMPLE PROGRAM OUTPUTS

```
What file next?
            LAN SIMULATION
*****************************************

INITIAL CONDITIONS
Quitting point                                     2000
Number of nodes                                       1
Distance between nodes in feet =                   1000
Bits per page =                                   16000
Pages per second =                                 1.00
Bits per packet =                                 12144
Propagation factor =                               0.68
Transmission speed in bits per second      1.000000e+07
Speed of light in feet per second          9.833894e+08

CALCULATED CONDITIONS
Data in packets per second =                 1.31752300
Slot time in seconds =                       0.00121440
Propagation delay in seconds =               0.00123141



TIME COMPONENTS OF SIMULATION
Number of Nodes =  1
Total operating time (seconds)             1518.00012207
Total channel usage (seconds)              1526.28759766
Average idle time (seconds)                   0.76300961
Average busy time (seconds)                   0.00243028
Average usage time w/o conflicts (sec)        0.00121440
Total idle time (seconds)                  1521.44116211
Total busy channel time (seconds)             4.84598541
Total channel usage time w/o conflicts        2.42151380


RESULTS OF SIMULATION
Number of nodes talking 1
Theoritical throughput (ST)                      0.0016
Simulated throughput (SS)                        0.0016
Total number of collisions                       6
Offered load in packets/slot   (G)               0.0016
LAN efficiency factor (SS/G)                     0.9916
```

```
What file next?
            LAN SIMULATION
*****************************************

INITIAL CONDITIONS
Quitting point                                        2000
Number of nodes                                         24
Distance between nodes in feet =                      1000
Bits per page =                                      16000
Pages per second =                                    0.30
Bits per packet =                                    12144
Propagation factor =                                  0.68
Transmission speed in bits per second         1.000000e+07
Speed of light in feet per second             9.833894e+08

CALCULATED CONDITIONS
Data in packets per second =                    0.39525694
Slot time in seconds =                          0.00121440
Propagation delay in seconds =                  0.00123141



TIME COMPONENTS OF SIMULATION
Number of Nodes =  24
Total operating time (seconds)                210.83332825
Total channel usage (seconds)                 215.04675293
Average idle time (seconds)                     0.10656445
Average busy time (seconds)                     0.00243028
Average usage time w/o conflicts (sec)          0.00121440
Total idle time (seconds)                     210.25166321
Total busy channel time (seconds)               4.79494953
Total channel usage time w/o conflicts          2.39601135


RESULTS OF SIMULATION
Number of nodes talking 24
Theoritical throughput (ST)                         0.0114
Simulated throughput (SS)                           0.0111
Total number of collisions                          27
Offered load in packets/slot    (G)                 0.0115
LAN efficiency factor (SS/G)                        0.9672
```

```
What file next?
            LAN SIMULATION
*****************************************

INITIAL CONDITIONS
Quitting point                                     2000
Number of nodes                                       1
Distance between nodes in feet =                   1000
Bits per page =                                 1228800
Pages per second =                                 0.30
Bits per packet =                                 12144
Propagation factor =                               0.68
Transmission speed in bits per second      1.000000e+07
Speed of light in feet per second          9.833894e+08

CALCULATED CONDITIONS
Data in packets per second =                30.35573196
Slot time in seconds =                       0.00121440
Propagation delay in seconds =               0.00123141



TIME COMPONENTS OF SIMULATION
Number of Nodes =  1
Total operating time (seconds)              65.88541412
Total channel usage (seconds)               68.45488739
Average idle time (seconds)                   0.03294696
Average busy time (seconds)                   0.00243028
Average usage time w/o conflicts (sec)        0.00121440
Total idle time (seconds)                   63.75236893
Total busy channel time (seconds)             4.70259857
Total channel usage time w/o conflicts        2.34986401


RESULTS OF SIMULATION
Number of nodes talking 1
Theoritical throughput (ST)                      0.0356
Simulated throughput (SS)                        0.0343
Total number of collisions                       65
Offered load in packets/slot   (G)               0.0369
LAN efficiency factor (SS/G)                     0.9312
```

```
What file next?
            LAN SIMULATION
*****************************************

INITIAL CONDITIONS
Quitting point                                     2000
Number of nodes                                      24
Distance between nodes in feet =                   1000
Bits per page =                                 1228800
Pages per second =                                 0.10
Bits per packet =                                 12144
Propagation factor =                               0.68
Transmission speed in bits per second      1.000000e+07
Speed of light in feet per second          9.833894e+08

CALCULATED CONDITIONS
Data in packets per second =                10.11857700
Slot time in seconds =                       0.00121440
Propagation delay in seconds =               0.00123141



TIME COMPONENTS OF SIMULATION
Number of Nodes =  24
Total operating time (seconds)               8.23567677
Total channel usage (seconds)                9.85358238
Average idle time (seconds)                   0.00410825
Average busy time (seconds)                   0.00243028
Average usage time w/o conflicts (sec)        0.00121440
Total idle time (seconds)                    6.19113016
Total busy channel time (seconds)             3.66243792
Total channel usage time w/o conflicts        1.83010089

RESULTS OF SIMULATION
Number of nodes talking 24
Theoritical throughput (ST)                      0.2276
Simulated throughput (SS)                        0.1857
Total number of collisions                       493
Offered load in packets/slot   (G)               0.2949
LAN efficiency factor (SS/G)                     0.6298
```

APPENDIX E

ASRM ACRONYMS

AIMS	Automated Information Management System
AMS	Automated Manufacturing System
API	Automated Process Instructions
ARPA	Advanced Research Project Agency
ARTS	Automated Requirements Traceability System
ASB	AeroJet Space Boosters
ASC	Area Supervisory Computer
ASE	Application Service Element
ASPC	AeroJet Solid Propulsion Company
ASRM	Advanced Solid Rocket Motor
ATP	Acceptance Test Procedure
AU	Attachment Unit
BAS	Building Automation System
BIS	Business Information System
BOM	Bill of Materials
CA	Certification Authority
CCB	Configuration Control Board
CI	Calibration Instructions
CLNP	Connectionless-mode Network Protocol
CMIS	Contract Management Information System
CMS	Configuration Management System
CNC	Computer Numerical Control
COS	Corporation for Open Systems
COTS	Commercial Off-the-Shelf
CRP	Capacity Requirements Planning
CUI	Common User Interface
CVIA	Computer Virus Industry Association
DA	Destination Address
DAC	Discretionary Access Control
DCA	Defense Communications Agency
DCS	Distributed Control System
DM	Document Management
DMANDS	Documentation Management & Distribution System

DR	Discrepancy Report
DWG	Design Working Group
ECMA	European Computer Manufacturers Association
ECRA	Engineering Change Request Approval
EP	Explosion Proof
EV	Enforcement Vector
FA	Final Assembly
FIMS	Facility Information Management System
FIPS	Federal Information Processing Standards
FNB	Forward Nozzle Bonding
FSN	Final Sequence Number
FTAM	File Transfer and Access Management
FTP	File Transfer Protocol
GOSIP	Government OSI Profile
IAN	Institutional Area Network
ICD	Interface Control Document
ICV	Integrity Check Value
IESS	Integrated Engineering Support System
IGDS	Interactive Graphics Design System (Intergraph protocol)
IGES	Initial Graphics Exchange Specification
IH	Industrially Hardened
IPI	Integrated Process Instruction(s)
IPMS	Integrated Project Management System
ITE	Integrated Telephone Equipment
ITI	Integrated Test Instruction(s)
KBS	Knowledge Based System
KMC	Key Management Center
KMP	Key Management Protocol
KMAE	Key Management Application Entity Protocol
LAD	Lockheed Austin Division
LAI	Laboratory Analysis Instruction
LAVA	Los Alamos Vulnerability Analysis
LI	Length Indicator
LIMS	Laboratory Information Management System
LLCSS	Logical Link Control Security Sublayer
LMSC	Lockheed Missiles & Space Company, Inc.
MAC	Media (or Mandatory) Access Control

MAP	Manufacturing Area Protocol
MBOM	Manufacturing BOM
MCMS	Manufacturing Configuration Management System
ME	Manufacturing Engineer
MHS	Message Handling System (office) - X400
MIB	Management Information Base
MIS	Management Information System
MNA	Marshall (MSFC) Network Architecture
MPS	Master Production Schedule
MRB	Material Review Board (or) Material Requirements Planning
MRP	Manufacturing Resource Planning
MSDS	Material Safety Data Sheet
MTA	Mail Transfer Agent
MTI	Morton Thiokol, Inc.
NCSC	National Computer Security Center
NDE	Non–Destructive Evaluation
NFM	Network File Management
NIST	National Institute of Standards & Technology (formerly NBS)
NR	Non–conformance Reports
NS	Network Service
NSA	National Security Agency
NSAP	Network Service Access Point
NSDU	Network Service Data Unit
ODA	Office Document Architecture
ODIF	Office Document Interchange Format
ODP	Office Data Protocol
OIS	Operations Information System
OSI	Open Systems Interconnect
OSS	Operations Support System
PAA	Peer Access Authorization
PAE	Peer Access Enforcement
PCS	Process Control Subsystem
PDU	Protocol Data Unit
PIPL	Program Indentured Parts List
PKCS	Public Key Crypto System
PLC	Programmable Logic Controller
PMMS	Performance Measurement Management System

PMS	Project Management System
PSCN	Program Support Communications Network (NASA Wide Network)
PSCN–I	PSCN Internet
QA	Quality Assurance
QASL	Quality Approval Supplier List
QOS	Quality of Service
RAR	Receiving Acceptance Reports
RID	Review Item Discrepancy
RII	Receiving Inspection Instruments
SA	Source Address
SDAR	Supplier's Discrepancy Action Request
SDNS	Secure Data Network(s) System
SE	Secure Entity
SE&I	System Engineering & Integration
SFC	Shop Floor Control
SILS	Standard for Inter–operable LAN Security
SMAE	Security Management Application Entity
SMIB	Security Management Information Base
SMTP	Simple Mail Transfer Protocol
SN	Subnetwork
SNICP	Subnetwork Independent Convergence
SOW	Statement of Work
SP	Security Protocol
SP3	Security Protocol 3
SP4	Security Protocol 4
SPC	Statistical Process Control
SQE	Supplier Quality Engineer
SQR	Supplier Quality Representative
SRM&QA	Safety, Reliability, Maintainability, and Quality Assurance
SRS	Software Requirements Specification
SSI	Source Surveillance Instruction
STE	Standard Telephone Equipment
TCSEC	Trusted Computer System Evaluation Criteria
TDC	Test Data Center
TEK	Traffic Encryption Key
TIS	Trusted Information Systems, Inc. (bought IBM's Secure Xenix)
TNI	Trusted Network Interpretation

TOP	Business
TSAP	Transport Service Access Point
UA	User Agent
UT	Ultrasonic Test
WBS	Work Breakdown Structure
YCCC	Yellow Creek Cad Coordinator
802.2	Logical Link Control (LLC)
802.3	CSMA/CD
802.4	Token Bus
802.5	Token Ring